Imagine you are tasked with designing a resilient architecture for a new online marketplace that will be launched soon. The marketplace will allow users to buy and sell various products, and will be accessed through a website and a mobile app.

Your goal is to design an architecture that is highly available and can handle a large volume of traffic and transactions, while also being fault-tolerant and scalable.

Identify the critical components of the system: Begin by identifying the critical components of the system that need to be highly available. These may include the website, mobile app, payment gateway, database, and other services that are essential for the marketplace to function.

Use microservices architecture: Use a microservices architecture to break down the system into smaller, independent services. Each service should be designed to be fault-tolerant and scalable, so that it can handle traffic and transactions even if other services fail.

Use load balancing and auto-scaling: Use load balancing and auto-scaling techniques to distribute traffic evenly across the services and to automatically add or remove resources based on demand.

Implement redundancy and failover mechanisms: Implement redundancy and failover mechanisms to ensure that the system can continue to function even if individual components fail. This may include using multiple servers, data centers, or cloud regions to ensure that there is always a backup available.

Use caching and content delivery networks (CDNs): Use caching and CDNs to improve the performance and availability of the system. This can help reduce latency and improve response times for users, while also reducing the load on the servers.

Monitor and test the system: Implement monitoring and testing tools to ensure that the system is performing as expected and to identify any potential issues or bottlenecks. This can help you identify and fix issues before they affect the users.